

CSS532 HW3 Report

Name: Iris(Qiaoyu) Zheng

1 Feature Realization :

a).Create an AWS Greengrass group

Set up your Greengrass Group

Setting up your Group requires you to provision a Core device in the IoT Registry, acquire a certificate for your Core, and assign an IAM role to your Group. If you're unfamiliar with any of these steps we recommend the easy Group creation. Finally, you'll need to install Greengrass software on your Core device.

Easy Group creation (recommended)

This process will automatically provision a Core in the registry, use default settings to generate a new Group, and provide your Core with a new certificate and a key pair.

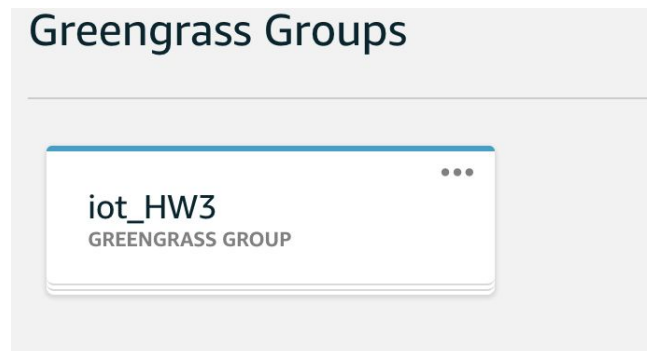
Use easy creation

Advanced Group creation

This customizable process will take you step-by step through the Core provisioning and will allow you to customize the IAM Role for your Group and the certificate for your Core, and provide a key pair.

Customize

CancelUse easy creation



Name the group as `iot_HW3`

b). Set up an EC2 micro instance as an AWS Greengrass Core

Launch Instance

Connect


Actions

Filter by tags and attributes or search by keyword

<input type="checkbox"/>	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
<input type="checkbox"/>		i-094c1a27156d21c52	a1.medium	us-west-2a	<div>running</div>	<div>2/2 checks</div>

Click launch Instance on EC2 page, customize the system as Ubuntu 64-bit(Arm), for future utilization with raspberry pi.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes


Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-06d51e91cea0dac8d (64-bit x86) / ami-02cbcd67225579b2c (64-bit Arm)

Free tier eligible
 Ubuntu Server 18.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select

☒ 64-bit (x86)
 ☐ 64-bit (Arm)

▼ Security Groups Edit security groups

Security group name	Description
launch-wizard-5	launch-wizard-5 created 2019-11-09T21:14:44.850-08:00

Type	Protocol	Port Range	Source	Description
------	----------	------------	--------	-------------

Do customization, especially, add MQTT protocol into the security group.

Filter by tags and attributes or search by keyword

1 to 1 of 1

	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
		i-094c1a27156d21c52	a1.medium	us-west-2a	running	2/2 checks ...	None	ec2-34-215-163-208.us...

Successfully launch the ex2 instance

Use the certification given when registering the instance to log into the ec2 instance, use the certification given when registering the core to install the Greengrass core software package. Install the AWS Greengrass Core SDK on this EC2 instance terminal and run the Daemon

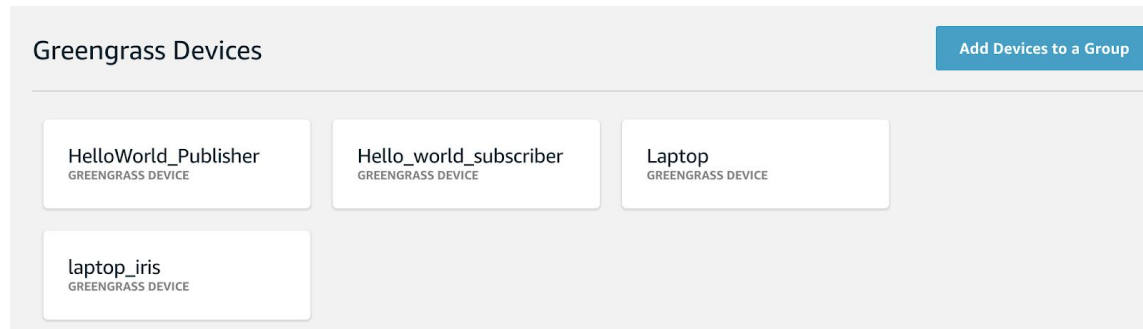
```

ubuntu@ip-172-31-53-203:/greengrass$ cd certs/
ubuntu@ip-172-31-53-203:/greengrass/certs$ ls
579ffac3ad.cert.pem  579ffac3ad.private.key  579ffac3ad.public.key  README  root.ca.pem
ubuntu@ip-172-31-53-203:/greengrass/certs$ vim 579ffac3ad.public.key
ubuntu@ip-172-31-53-203:/greengrass/certs$ cd /greengrass/ggc/core/
ubuntu@ip-172-31-53-203:/greengrass/ggc/core$ sudo ./greengrassd start
Setting up greengrass daemon
Validating hardlink/softlink protection
Waiting for up to 1m10s for Daemon to start

greengrass successfully started with PID: 3485
ubuntu@ip-172-31-53-203:/greengrass/ggc/core$ ps aux | grep 3485
root      3485  0.3  0.7 463400 15092 pts/0    S1   08:16   0:00 /greengrass/ggc/packages/1.9.4/bin/daemon -core-dir /greengrass/ggc
/packages/1.9.4 -greengrassdPid 3482
ubuntu    3504  0.0  0.0   6100   644 pts/0    S+   08:17   0:00 grep --color=auto 3485
  
```

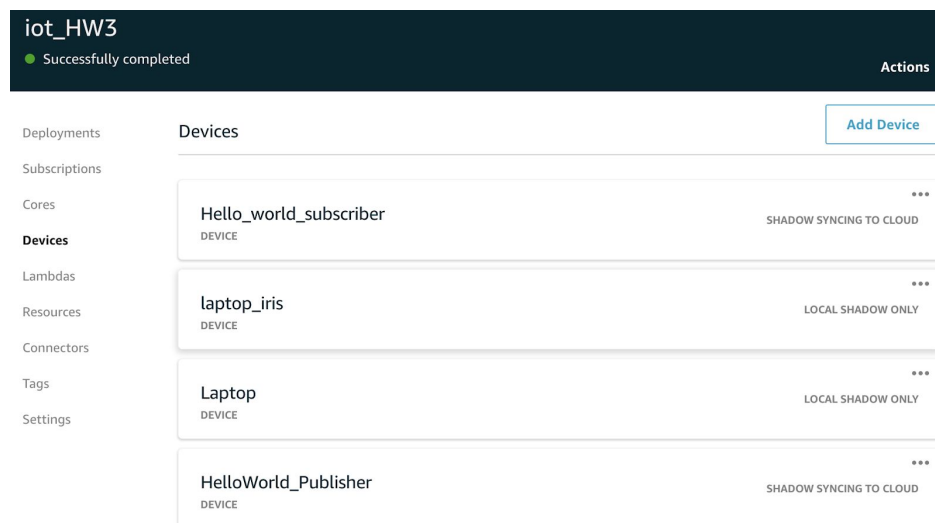
c). Use the laptop as an AWS IoT device.

Register the devices in the Greengrass group as the thing and install AWS IoT SDK on the laptop and install AWS IoT SDK on this device.



d). Add the core and IoT devices to the Greengrass group. Notice that you need to register your core and IoT device at first, then configure them (in AWS console) as core and IoT devices accordingly.

Register and add the core and devices to the IoT the Greengrass group.

A screenshot of the 'Add a Device' dialog box in the AWS IoT Greengrass console. The dialog has a blue header with the title 'Add a Device'. Below the header, there is a paragraph of text: 'Greengrass Devices can be created by re-purposing an existing IoT Thing from your Registry or by creating new Registry items, and then adding them to a Greengrass Group.' There are two main sections. The first section is titled 'Create a new Device' and includes the text 'You will create a new Device and generate a certificate, a private key and a public key.' To the right of this text is a button labeled 'Create New Device'. The second section is titled 'Use an existing IoT Thing as an Device' and includes the text 'You can add an existing IoT Thing to your Group.' To the right of this text is a button labeled 'Select an IoT Thing'. At the bottom of the dialog, there are three buttons: 'Cancel', 'Back', and 'Create New Device'.

iot_HW3

Successfully completed

Actions

Deployments

Subscriptions

Cores

Devices

Lambdas

Resources

Connectors

Tags

Settings

Cores

iot_HW3

CORE

e). Develop and deploy a Lambda function named "LambdaCore" to the Greengrass Core

Create a Lambda function, zip python file, and Greengrass SDK and upload it to the lambda function. Set the lambda function in the Greengrass group page

File Edit Find View Go Tools Window

Environment

test_receive.py

23 message = event['test-key']
24 except Exception as e:
25 logging.error('Message could not be parsed. ' + repr(e))
26 return message
27
28 def function_handler(event, context):
29 try:
30 input_topic = get_input_topic(context)
31 input_message = get_input_message(event)
32 response = 'Invoked on topic "%s" with message "%s"' % (input_topic, input_message)
33 response = response + my_platform
34 logging.info(response)
35 except Exception as e:
36 logging.error(e)
37
38 client.publish(topic=OUTPUT_TOPIC, payload=response)
39
40 storeData = {}
41 storeData['weatherRecord'] = event['test-key']
42 jsonData = json.dumps(storeData)

iot_HW3

Successfully completed

Actions

Deployments

Subscriptions

Cores

Devices

Lambdas

Resources

Connectors

Tags

Settings

Lambdas

Greengrass_Helloworld

LAMBDA FUNCTION

USING ALIAS: GG_HELLOWORLD

send_message

LAMBDA FUNCTION

USING V1

send_message_console

LAMBDA FUNCTION

USING V3

Add Lambda

Deployments	Subscriptions			Add Subscription
Subscriptions	Source	Target	Topic	
Devices	IoT Cloud	send_message_console	hello/world	...
Lambda	IoT Cloud	send_message_console	test/input_message	...
Resources	send_message_console	IoT Cloud	test/topic_results	...
Connectors	send_message_console	Laptop	hello/world	...

Create subscription between devices(laptop), lambda function and the IoT console.

GREENGRASS GROUP
iot_HW3
Successfully completed

Deployments
Subscriptions
Cores
Devices

Group history overview
By deployment

Deploy
Delete Group
Reset Deployments

Deployed	Version	Status
Nov 9, 2019 7:58:48 PM -0800	142480a5-3c00-420c-9f3c-5a7aa8e88d61	Successfully complet... ...

Run IoT Greengrass Daemon on the ec2 instance and deploy it.

Run the python file sending the Message: Current weather and the console can receive the data. Data is stored into the s3 via the lambda function.

test/topic_results
Nov 9, 2019 10:52:57 PM -0800
Export Hide

We cannot display the message as JSON, and are instead displaying it as UTF-8 String.

```

Invoked on topic "hello/world" with message "Current weather is: Rainy"Linux-
4.15.0-1052-aws-aarch64-with-Ubuntu-18.04-bionic

```

Type a prefix and press Enter to search. Press ESC to clear.

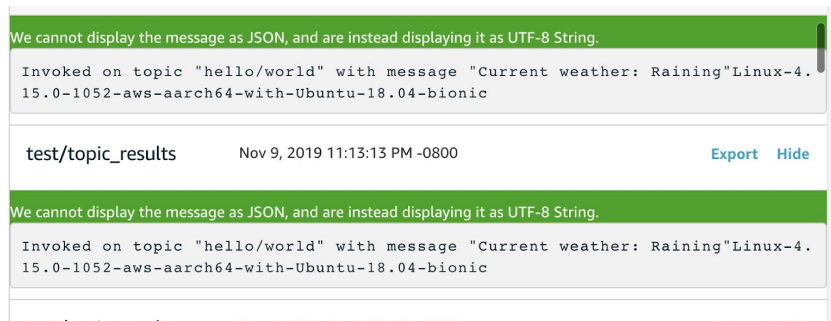
Upload
Create folder
Download
Actions

☐ Name

☐ data.json

e) Develop a program running on the IoT device, the code is shown in the folder.

```
2019-11-09 23:06:03,629 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
2019-11-09 23:06:03,629 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Offline request detected!
2019-11-09 23:06:03,629 - AWSIoTPythonSDK.core.protocol.internal.queues - DEBUG - append: Add new element: <AWSIoTPythonSDK.core.pro
tocol.internal.requests.QueueableRequest object at 0x10576fcc0>
Published topic hello/world: {"test-key": "Current weather: Raining", "sequence": 18}
```



2 Problems Solving :

a). Install IoT Greengrass software package in ec2 instance

It took me a lot of time to configure the instance and install the correct version of software according to the system chosen for the ec2 instance. Also, authentication failed at the beginning, since the Root CA I used is AmazonRootCA3.pem at the beginning, but Greengrass only works well with AmazonRootCA1.pem.

b). Link to the core

I am a kind of confused since I am not able to link to the core via private IP address. The linking problem is fixed after I set up the connection manually and use the public IP address, but I am still trying to understand why I can only use public IP but most of the examples I see use private IP.

c). Deploy Failure

Deploy failed every time I update the lambda function version. The problem is fixed by deleting and recreating subscription. But I am still trying to find a more "reasonable" way to updating the lambda function without impacting the subscribing.

3 Time Consuming

Around 12 hours